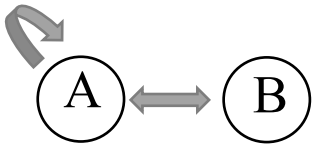
2. Nicholas

Program Name: Nicholas.java **Input File: nicholas.dat**

Nicholas is learning about how to manually count path lengths in a graph, such as the one shown here, using adjacency matrices, but still needs your help to confirm his counting. He started with this simple example, which produces the following adjacency matrix:

```
  A B
A 1 1
B 1 0
```



He has learned that this matrix means that given two vertices A and B in the graph above, there is a connection from A back to itself, and a two-way connection from A to B. To count the number of paths of length one, or direct connections in the graph, all he has to do is count the number of 1s in the graph, three in this case, represented in letter notation as AA, AB, and BA. AA means that the connection starts and ends at A, AB means it starts at A and ends at B, and so on.

However, counting the number of two-hop paths is a little more involved. He can see that AAA is a possibility, and then ABA, and then BAB, but wonders if AA could be combined with B. His teacher says, "Yes, it can.", so he adds AAB and BAA to his count, making a total of five 2-hop paths.

"What about 3-hop paths?", he wonders. Hmmm, let's see, starting from A there would be AAAA, AAAB, AABA, ABAA, and ABAB. Starting from B, the 3-hop paths are BAAA, BAAB, and BABA. Altogether, that would make eight 3-hop paths within this graph.

Your job is to take any graph with at least two, but no more than five vertices, and help Nicholas confirm his counting of the number of paths of any length from 1 to 3 hops.

Input: Several lines of data, each consisting of an integer N, indicating a graph with N nodes, followed by an integer P, the path length to calculate, followed by N strings, each of length N, containing 0s and 1s, indicating the adjacency matrix for the graph.

Output: The number of P length paths in the given matrix.

Sample Input:

```
2 1 11 10
2 2 11 10
2 3 11 10
3 1 111 001 001
4 2 0100 0010 0001 1000
```

Sample Output:

```
3
5
8
5
4
```